IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: BRIAN TRAN ET AL.

Serial No.:	10/709,415	Art Unit:	2166
Filed:	5/04/2004	Examiner:	Johnese Johnson
Title:	*Self Adaptive Prefix Encoding for Stable Node Identifiers*		

REQUEST FOR REINSTATEMENT OF APPEAL

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Commissioner:

Applicants filed an Appeal Brief on 09/18/2007, which resulted in the reopening of prosecution and issuance of an Office Action on 12/14/2007. As per 37 C.F.R. 1.111, Applicants hereby respectfully request reinstatement of appeal with respect to the non-final office action dated 12/14/2007. Applicants submit the below attached Supplemental Appeal Brief in response to the non-final office action dated 12/14/2007.

Respectfully submitted by
Applicant's Representative,

/ramraj soundararajan/

Ramraj Soundararajan
Reg. No. 53,832

IP AUTHORITY, LLC
4821A Eisenhower Ave
Alexandria, VA 22304
(703)461-7060

April 14, 2008

Serial No. 10/709,415
Group Art Unit 2166
Docket No: SVL920030099US1

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

SUPPLEMENTAL APPEAL BRIEF – 37 C.F.R § 1.192

U.S. Patent Application 10/709,415 entitled:

"Self-Adaptive Prefix Encoding for Stable Node Identifiers"

Real Party in Interest: International Business Machines Corporation

Related Appeals and Interferences:

An appeal brief was previously filed on 09/18/2007, which resulted in the reopening of prosecution and the subsequent issuance of an office action dated 12/14/2007.

Status of Claims:

Claims 1-16 are pending.

Claims 1-16 are rejected.

Claims 1-4, 9 and 14-16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hu (U.S. Patent 7,274,671), in view of Bunton (U.S. Patent 5,151,697), and further in view of O'Neil (U.S. Patent 6,889,226).

Claims 1-16 are hereby appealed.

Status of Amendments:

An amendment was filed on 3/14/2008 in response to the Office Action of 12/14/2007. The amendment corrected a minor objection pointed out by the Examiner in the Office Action of 12/14/2007 with respect to claim 9. This amendment was done without adding new matter.

Summary of Claimed Subject Matter:

(NOTE: All citations are made from the associated Pre-Grant Publication 2006/0004858)

According to independent **claim 1**, the present invention provides for a computer-based method comprising the steps of: (a) choosing an initial base length with which to encode local identifiers ***(see paragraphs [0034], [0035], [0038], and [0039])***, (b) assigning a value of zero as

a node identifier to a root node in a logical tree ***(see paragraph [0032] and figures 1-5)***, (c) sequentially assigning to descendants of a root node a local identifier having an even value and a length equal to said base length chosen in said choosing step, wherein said local identifiers are assigned in increasing value from leftmost children to rightmost children ***(see paragraph [0032] and figures 1-5)***, (d) assigning node identifiers by concatenating local identifiers of all nodes along a path from a root node to a node to which a node identifier is currently being assigned ***(see paragraphs [0032], [0033] and figures 1-5)***, and (e) extending said initial base length when local identifier encoding combinations are exhausted before all descendants are assigned local identifiers ***(see paragraphs [0034], [0035], [0036], [0037], [0038] and [0039] and figures 1-5)***.

According to independent **claim 9**, the present invention provides for an article of manufacture, said article of manufacture comprising a computer readable storage medium having computer readable program code embodied therein, said computer readable program code comprising modules being executed by a computer comprising modules implementing code to: (a) choose an initial base length with which to encode local identifiers ***(see paragraphs [0034], [0035], [0038], [0039] and [0049])***, (b) assign a value of zero as a node identifier to a root node in a logical tree ***(see paragraphs [0032] and [0049] and figures 1-5)***, (c) sequentially assign to descendants of a root node a local identifier having an even value and a length equal to said base length chosen in said choosing step, wherein said local identifiers are assigned in increasing value from leftmost children to rightmost children ***(see paragraphs [0032] and [0049] and figures 1-5)***, (d) assign node identifiers by concatenating local identifiers of all nodes along a path from a root node to a node to which a node identifier is currently being assigned ***(see paragraphs [0032], [0033], and [0049] and figures 1-5)***, and (e) extend said initial base length

when local identifier encoding combinations are exhausted before all descendants are assigned local identifiers ***(see paragraphs [0034], [0035], [0036], [0037], [0038] and [0039] and figures 1-5)***.

According to independent **claim 16**, the present invention provides for a computer-based method comprising the steps of: (a) choosing an initial base length with which to encode local identifiers ***(see paragraphs [0034], [0035], [0038], [0039] and [0049])***, (b) assigning a value of zero as a node identifier to a root node in a logical tree ***(see paragraphs [0032] and [0049] and figures 1-5)***, (c) sequentially assigning to descendants of a root node a local identifier having an even value and a length equal to said base length chosen in said choosing step, wherein said local identifiers are assigned said even values based on variable-length binary string encoding and said local identifiers are assigned in increasing value from leftmost children to rightmost children ***(see paragraphs [0032], [0039], and [0049] and figures 1-5)***, (d) assigning node identifiers by concatenating local identifiers of all nodes along a path from a root node to a node to which a node identifier is currently being assigned ***(see paragraphs [0032], [0033], and [0049] and figures 1-5)***, and (e) extending said initial base length when local identifier encoding combinations are exhausted before all descendants are assigned local identifiers ***(see paragraphs [0034], [0035], [0036], [0037], [0038] and [0039] and figures 1-5)***.

Grounds of Rejection to be Reviewed on Appeal:

Claims 1-16 are hereby appealed.

1. Was an ambiguous rejection provided under 35 U.S.C. §103(a) with respect to claims 5-8 and 10-13?

2. Claims 1-4, 9 and 14-16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hu (U.S. Patent 7,274,671), in view of Bunton (U.S. Patent 5,151,697), and further in view of O'Neil (U.S. Patent 6,889,226). Was a proper rejection made under 35 U.S. C. §103(a) using existing USPTO guidelines?

ARGUMENT:

1. Was an ambiguous rejection provided under 35 U.S.C. §103(a) with respect to claims 5-8 and 10-13?

The Examiner in page 3 of the Office Action dated 12/14/2007 (Under 'Claim Rejections - 35 U.S.C. §103' – Bullet Point #6) states that "Claims 1-4, 9, and 14-16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hu (U.S. Pat. No. 7,274,671), in view of Bunton (U.S. Pat. No. 5,151,697), and further in view of O'Neil (U.S. Pat. 6,889,226)." Absent from the Examiner's 35 U.S.C. §103 rejection is a status listing for claims 5-8 and 10-13 under Bullet Point #5. However, on pages 6-13 of the Office Action dated 12/14/2007, and under same Bullet Point #5, the Examiner has outlined rejections for claims 5-8 and 10-13, which appear to address a 35 U.S.C. §103 rejection using Hu, O'Neil, and Bunton. Applicants respectfully assert that proper M.P.E.P. procedures were not followed in the rejection of claims 5-8 and 10-13 as there is no proper summary statement in the Office Action dated 12/14/2007 with regards to what references were used in the 35 U.S.C. §103 rejection. Based on the Examiner's statements on pages 6-13, it is assumed that the Examiner has used Hu, O'Neil, and Bunton in the rejections of claims 5-8 and 10-13. The combination of these references has been argued below with respect to at least the independent claims. However, if the Examiner intended to use references other than Hu, O'Neil or Bunton in the rejection of claims 5-8 and 10-13, Applicants reserve the right to present additional arguments if the Examiner were to provide further clarification with respect to claims 5-8 and 10-13.

2. <u>Was a proper rejection made under 35 U.S. C. §103(a) using existing USPTO guidelines?</u>

Claims 1-4, 9 and 14-16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hu (U.S. Patent 7,274,671), in view of Bunton (U.S. Patent 5,151,697), and further in view of O'Neil (U.S. Patent 6,889,226). To be properly rejected under 35 U.S.C. 103(a), the combination of references (i.e., Hu, Bunton, and O'Neil) must teach all the features of the rejected claims. Applicants respectfully submit that the combination of Hu, Bunton, and O'Neil fail to teach many of the features of Applicants' pending claims.

Hu provides for a technique for bitwise adaptive encoding wherein the technique leverages the frequency of an escape symbol for better compression.

Bunton teaches an adaptive data compression scheme that uses a fixed size dictionary.

O'Neil provides for a technique to represent hierarchical data in a non-hierarchical data structure, wherein the hierarchical data (e.g., XML data) can be viewed as having a "tree" structure, and each node in the tree is assigned a position identifier that represents both the depth level of the node within the hierarchy, and its ancestor/descendant relationship to other nodes.

Applicants' claim 1, by stark contrast, provides for a computer-based method comprising the steps of: (a) choosing an initial base length with which to encode local

identifiers, (b) assigning a value of zero as a node identifier to a root node in a logical tree, (c) sequentially assigning to descendants of a root node a local identifier having an even value and a length equal to said base length chosen in said choosing step, wherein said local identifiers are assigned in increasing value from leftmost children to rightmost children, (d) assigning node identifiers by concatenating local identifiers of all nodes along a path from a root node to a node to which a node identifier is currently being assigned, and (e) extending said initial base length when local identifier encoding combinations are exhausted before all descendants are assigned local identifiers.

Applicants' claim 9 provides for an article of manufacture implementing the above-described method.

Applicants' claim 16, also by stark contrast, provides for a computer-based method comprising the steps of: (a) choosing an initial base length with which to encode local identifiers, (b) assigning a value of zero as a node identifier to a root node in a logical tree, (c) sequentially assigning to descendants of a root node a local identifier having an even value and a length equal to said base length chosen in said choosing step, wherein said local identifiers are assigned said even values based on variable-length binary string encoding and said local identifiers are assigned in increasing value from leftmost children to rightmost children, (d) assigning node identifiers by concatenating local identifiers of all nodes along a path from a root node to a node to which a node identifier is currently being assigned, and (e) extending said initial base length when local

identifier encoding combinations are exhausted before all descendants are assigned local identifiers.

With respect to independent claims 1, 9, and 16, the Examiner states on page 3 of the non-final office action dated 12/14/2007 that column 3, lines 3-4 of the Hu reference teaches the feature of "assigning a value of zero as a node identifier to a root node in a logical tree". Column 3, lines 3-4 of Hu are provided verbatim below:

> "If there are more than one bits that attain the maximum value, choose anyone of the bits as M. At the root (Layer 0) of the maximum disparity tree, the symbols are partitioned into two groups, the "Zero group" and the "One Group".

Applicants emphasize that the above citation merely states that the root belongs to "layer 0" or the top layer and make no mention of assigning a zero value to the node identifier of the root. For support, Applicants have also provided below Figure 3 of Hu which clearly shows the various layers during partitioning.

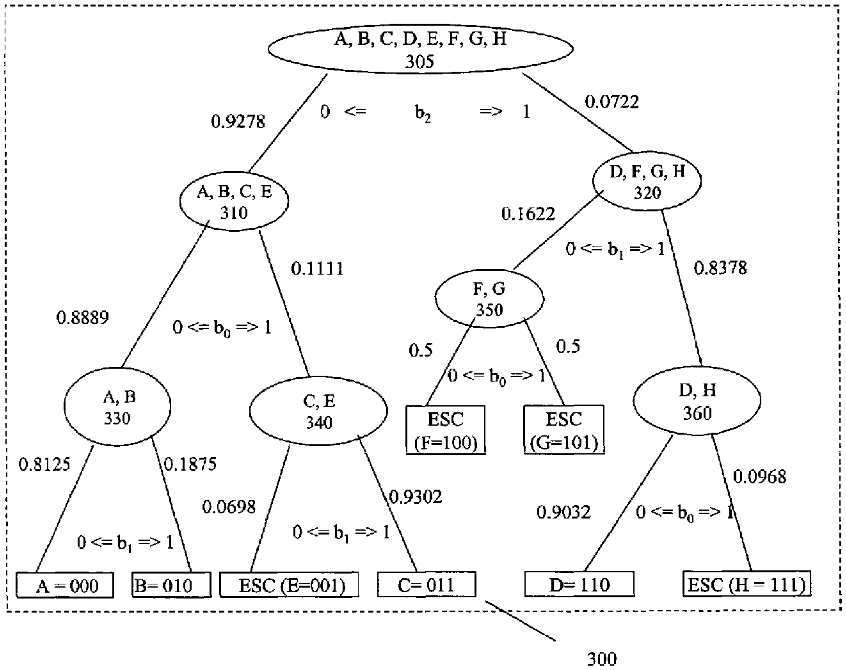

Figure 3 of Hu merely shows symbols A, B, C, D, E, F, G, and H being partitioned iteratively into a "zero group" and "one group". For example, symbols A,B are partitioned into a "zero group" comprising A and a "one group" comprising B. Further, probabilities such as the shown "0.8125" and "0.1875" are calculated based on probability values shown in table 210. For example, table 210 indicates that for a prefix of A, the probability that the next symbol is A is 65% and the probability that the next symbol is B is 15%. Therefore, when partitioning symbols A,B into A, a probability of (65)/(65+15)= 0.8125 is used. Likewise, when partitioning symbols A,B into B, a probability of (15)/(65+15)= 0.1875 is used. Figure 3 of Hu shows that based on such partitioning, the symbol group A, B, C, D, E, F, G, and H can be encoded using binary bits $b_2b_1b_0$, wherein A=000, B=010, C=011, D=110, E=001, F=100, G=101, and H=111 (see encoded values in boxes in Figure 3 of Hu and also presented in column 6, lines 5-14

of Hu).

Hence, it can be clearly seen from the data tree of Hu that the top most node in the tree is a set of all symbols (i.e., A, B, C, D, E, F, G, and H) and NOT, as the Examiner maintains, a root node with an assigned value of zero. In fact, the set of symbols (i.e., A, B, C, D, E, F, G, and H) are not assigned any values in Figure 3 of Hu, but merely represent the set that is used in partitioning.

Further, with respect to claims 1, 9, and 16, the Examiner states that Figure 3 of Hu teaches Applicants' feature of "sequentially assigning to descendants of a root node a local identifier having an even value and a length equal to said base length chosen in said choosing step, wherein said local identifiers are assigned in increasing value from leftmost children to rightmost children". Applicants respectfully disagree with the Examiner's conclusion.

As mentioned above, Figure 3 merely shows starting from a group of nodes (i.e., A, B, C, D, E, F, G, and H) and iteratively partitioning the group into a "zero group" and "one group" based on probabilities. For example, as seen in Figure 3 of Hu, the group (i.e., A, B, C, D, E, F, G, and H) is first partitioned into zero group A, B, C, E and one group D, F, G, H. Next, the group A, B, C, E is partitioned into zero group A,B and one group C,E. Similarly, group D, F, G, H is partitioned into zero group F,G and one group D,H. Group A,B is partitioned into zero group A and one group B (and subsequently

encoded as 000 and 010, respectively). Group C,E is partitioned into zero group E and one group C (and subsequently encoded as 001 and 011, respectively). Group F,G is partitioned into zero group F and one group G (and subsequently encoded as 100 and 101, respectively). Group D,H is partitioned into zero group D and one group H (and subsequently encoded as 110 and 111, respectively). Such iterative partitioning **CANNOT** be equated to Applicants' feature of sequentially assigning to descendants of a root node a local identifier having an even value and a length equal to a chosen base length, wherein said local identifiers are assigned in increasing value from leftmost children to rightmost children. Further, it should be emphasized that the iterative partitioning shown in Figure 3 of Hu fails to teach any assignation of identifiers to descendants of a root node.

It should be clear from the description above that Hu's Figure 3 merely teaches partitioning a bigger set into individual symbols based on probabilities and encoding such symbols using binary bit values and makes no mention of sequentially assigning, to descendants of a root node, a local identifier having an even value and a length equal to a chosen base length, wherein the local identifiers are assigned in increasing value from leftmost children to rightmost children.

With respect to claims 1, 9, and 16, the Examiner is correct in stating that Hu fails to teach the feature of "assigning node identifiers by concatenating local identifiers of all nodes along a path from a root node to a node to which a node identifier is currently

being assigned". However, Applicants respectfully disagree with the Examiner that this is remedied by the Bunton reference.

Specifically, the concatenation aspect mentioned in Bunton corresponds to the concatenation of characters while traversing a path from the root node to the related node, wherein each node has a character associated with it. Bunton's tree shown in Figure 3 is reproduced below:

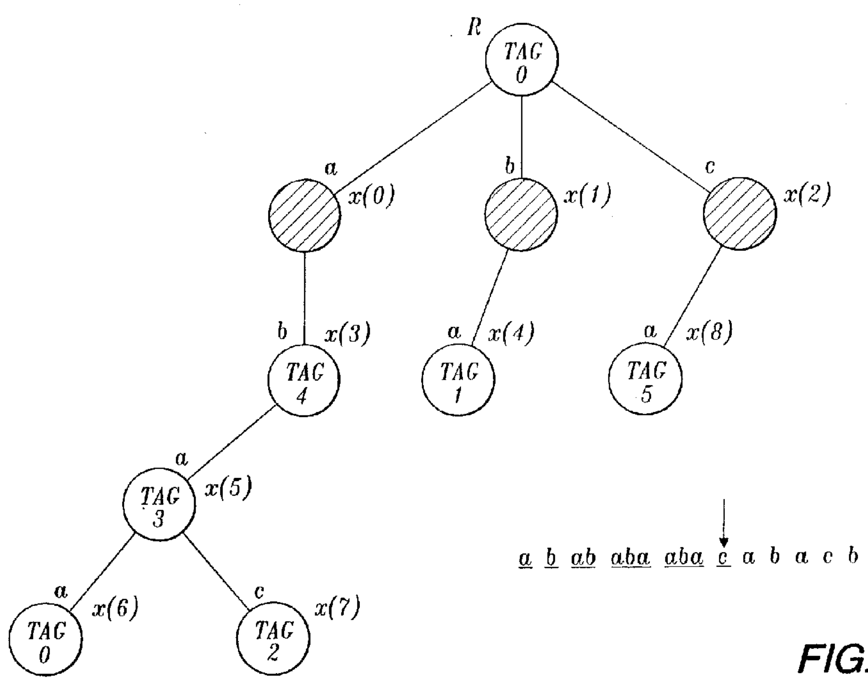

FIG.3.

Applicants wish to emphasize that in Figure 3 of Bunton, *x(0)*, *x(1)*, *x(2)*, *x(3)*, *x(4)*, *x(5)*, *x(6)*, *x(7)*, and *x(8)* represent *codes*, while a, b, and c represent alphabets. The concatenation that is taught in Bunton relates to the *concatenation of alphabets* and **NOT**, as the Examiner suggests, concatenation of codes. In fact, Applicants respectfully assert

that there is no teaching or suggestion for concatenating codes *x(0)*, *x(1)*, *x(2)*, *x(3)*, *x(4)*, *x(5)*, *x(6)*, *x(7)*, and *x(8)*.

For clarification of claims 1, 9, and 16, the Examiner is respectfully requested to review figure 4 of the application-as-filed, where Applicants' concatenation feature is outlined.

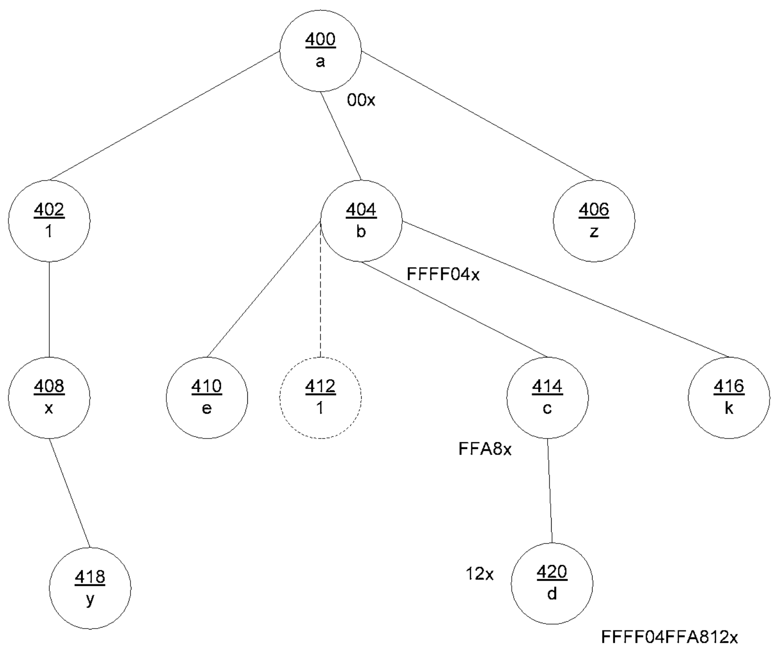

FIGURE 4

It can be seen from the above-figure that what is concatenated as per Applicants' invention is the encodings for nodes *a*, *b*, *c*, and *d*, whose values are 0, FFFF04x, and FFA8x, and 12x, respectively. Such a concatenation results in a node id for *d* with a value of FFFF04xFFA8x12x, which is a concatenation of the individual strings 0, FFFF04x, and FFA8x, and 12x. It should be emphasized that such concatenation is specifically used in the assignment of node identifiers and CANNOT be equated to a

<u>concatenation of alphabets</u>.

Applicants also wish to make note that the third reference used by the Examiner, O'Neil, fails to remedy the shortcomings of the Hu and Bunton references with respect to many of the features of independent claims 1, 9, and 16. Also, the previous Appeal Brief filed 09/18/2007 outlines many of the shortcomings associated with the O'Neil reference in detail. The Board of Patent Appeals and Interferences is hereby respectfully requested to review many of the arguments provided with regards to the O'Neil reference.

Applicants, therefore, submits that a prima facie case of obviousness has not been successfully established with respect to independent claims 1, 9, and 16. Hence, Applicants respectfully assert that an improper 35 U.S.C. §103 rejection was issued with regards to independent claims 1, 9, and 16 as the cited combination of Hu, Bunton and O'Neil references fail to teach or suggest many of the features of Applicants' pending claims 1, 9, and 16.

The arguments presented above substantially apply to dependent claims 2-8, 10-15 as they inherit all the features of the claim from which they depend. Hence, Applicants believe that dependent claims 2-8 and 10-15 are allowable over the cited art at least for the reasons set forth above. Hence, Applicants respectfully assert that an improper 35 U.S.C. §103 rejection was issued with regards to dependent claims 2-8 and 10-15.

SUMMARY

As has been detailed above, none of the references, cited or applied, provide for the specific claimed details of applicant's presently claimed invention, nor render them obvious. It is believed that this case is in condition for allowance and reconsideration thereof and early issuance is respectfully requested.

As this Appeal Brief has been timely filed within the set period of response, no fee for extension of time is required. However, the Commissioner is hereby authorized to charge any deficiencies in the fees provided, including extension of time, to Deposit Account No. 09-0460.

Respectfully submitted by
Applicant's Representative,

/ramraj soundararajan/

Ramraj Soundararajan
Reg. No. 53,832

IP AUTHORITY, LLC
4821A Eisenhower Ave
Alexandria, VA 22304
(703)461-7060

April 14, 2008

Claims Appendix:

1. (Previously Presented) A computer-based method comprising the steps of:

a. choosing an initial base length with which to encode local identifiers,

b. assigning a value of zero as a node identifier to a root node in a logical tree,

c. sequentially assigning to descendants of a root node a local identifier having an even value and a length equal to said base length chosen in said choosing step, wherein said local identifiers are assigned in increasing value from leftmost children to rightmost children,

d. assigning node identifiers by concatenating local identifiers of all nodes along a path from a root node to a node to which a node identifier is currently being assigned, and

e. extending said initial base length when local identifier encoding combinations are exhausted before all descendants are assigned local identifiers.

2. (Previously Presented) The computer-based method of claim 1, wherein inserting a node into an existing tree does not require change to existing node identifiers.

3. (Previously Presented) The computer-based method of claim 1, wherein a node is inserted between a first node and a second node having consecutive local identifiers.

4. (Previously Presented) The computer-based method of claim 3, wherein said inserted node is assigned a local identifier having a string length longer than string length of said first node.

5. (Previously Presented) The computer-based method of claim 1, wherein assigning said node identifier to an inserted node comprises the following steps:

a. determining whether node to be inserted is inserted as a first child, between two existing siblings, or as a last child under a single parent node,

b. when said node to be inserted is inserted as a first child under said single parent node,

i. checking last byte of an existing first child,

ii. when the value of said last byte is not the smallest even number, then an even number greater than zero and less than the value of said last byte is selected to generate a local identifier of said node to be inserted, else

iii. when the value of said last byte of an existing first child is the smallest even number, generating a local identifier for said node to be inserted by replacing said last byte of said existing first child by an odd number to generate a local identifier and extending node identifier of said existing first child by a byte having a value of any arbitrary even number,

c. when said node to be inserted is inserted between two existing siblings under said single parent node, determining whether the string length of node identifier of said first sibling is less than, equal to, or greater than the string length of node identifier of said second sibling, else

d. when said node to be inserted is inserted as a last child after all other children under said single parent node, assigning to said node to be inserted an even local identifier greater than that of existing last child under said single parent node, and

generating a node identifier by a concatenation of local identifiers of all nodes along a path from a root node to said node to be inserted.

6. (Previously Presented) The computer-based method of per claim 5, when said node to be inserted is inserted between two existing siblings under said single parent node and when the string length of local identifier of said first sibling is less than the string length of the local identifier of said second sibling,

a. checking when local identifier of said first sibling is the last available encoding value having a string length of the local identifier of said first sibling and being smaller in value than said local identifier of said second sibling,

b. when said local identifier of said first sibling is the last combination having a string length of the local identifier of said first sibling that is smaller in value than said local identifier of said second sibling,

i. when the local identifier of said second sibling is not the first available identifier having the string length of the local identifier of said second sibling that is greater than the value of said local identifier of said first sibling; an even-valued local identifier being less in value than said local identifier of said second sibling and having string length of local identifier of said second sibling is generated and assigned, else

ii. generating a local identifier for said node to be inserted by replacing said last byte of said existing first child by an odd number and extending local identifier of said existing first child by a byte having a value of any arbitrary even number less in value than said last byte of said existing first child, and

generating a node identifier by a concatenation of local identifiers of all nodes along a path from a root node to said node to be inserted.

7. (Previously Presented) The computer-based method of claim 5, when said node to be inserted is inserted between two existing siblings under said single parent node and when the string length of the local identifier of said first sibling is equal to the string length of the local identifier of said second sibling,

a. when the value of the local identifier of said first sibling plus two is less than the value of the local identifier of said second sibling, a local identifier for said node to be inserted takes on an even value greater than or equal to the value of said local identifier of first sibling plus two and less than the value of the local identifier of said second sibling,

b. when the string length of the local identifier of said first sibling plus two is equal to the string length of the local identifier of said second sibling, then the string length of the local identifier for said node to be inserted is extended wherein the length of the local identifier for the newly inserted node is the string length of said second sibling plus one, and the value of the first string length of said first sibling bytes is the node identifier of said first sibling plus one, and the new byte is an arbitrary even number less than the value of said last byte of the node identifier of said second sibling, and

generating a node identifier by a concatenation of local identifiers of all nodes along a path from a root node to said node to be inserted.

8. (Previously Presented) The computer-based method of claim 5, when said node to be inserted is inserted between two existing siblings under said single parent node and when the string length of the local identifier of said first sibling is greater than the string length of the local identifier of said second sibling

a. when the local identifier of said second sibling is not the smallest value having the string length of said second sibling that is greater in value than the local identifier of said first sibling, then a local identifier having a string length of said second sibling and having even value smaller than the value of the last byte of the node identifier of said second sibling is generated and assigned else,

b. when the local identifier of said first sibling is not the largest value with the string length of the local identifier of said first sibling, one of the larger values for the new encoding is generated and assigned, else

c. extending the local identifier of said first sibling by a length, by setting the last byte to the highest odd number and the new byte to an even number less than the value of the last byte, and

generating a node identifier by a concatenation of local identifiers of all nodes along a path from a root node to said node to be inserted.

9. (Previously Presented) An article of manufacture, said article of manufacture comprising a computer readable storage medium having computer readable program code embodied therein, said computer readable program code comprising modules being executed by a computer comprising modules implementing code to:

a. choose an initial base length with which to encode local identifiers,

b. assign a value of zero as a node identifier to a root node in a logical tree,

c. sequentially assign to descendants of a root node a local identifier having an even value and a length equal to said base length chosen in said choosing step, wherein said local identifiers are assigned in increasing value from leftmost children to rightmost children,

d. assign node identifiers by concatenating local identifiers of all nodes along a path from a root node to a node to which a node identifier is currently being assigned, and

e. extend said initial base length when local identifier encoding combinations are exhausted before all descendants are assigned local identifiers.

10. (Previously Presented) The article of manufacture of claim 9, wherein assigning a prefix encoded node identifier to an inserted node comprises modules implementing code to:

a. determine whether node to be inserted is inserted as a first child, between two existing siblings, or as a last child under a single parent node,

b. when said node to be inserted is inserted as a first child under said single parent node,

i. check last byte of an existing first child,

ii. when the value of said last byte is not the smallest even number, then an even number greater than zero and less than the value of said last byte is selected to generate a local identifier of said node to be inserted, else

iii. when the value of said last byte of an existing first child is the smallest even number, generate a local identifier for said node to be inserted by replacing said last byte of said existing first child by an odd number to generate a local identifier and extending node identifier of said existing first child by a byte having a value of any arbitrary even number,

c. when said node to be inserted is inserted between two existing siblings under said single parent node, determine whether the string length of node identifier of said first sibling is less than, equal to, or greater than the string length of node identifier of said second sibling, else

d. when said node to be inserted is inserted as a last child after all other children under said single parent node, assign to said node to be inserted an even local identifier greater than that of existing last child under said single parent node, and

generate a node identifier by a concatenation of local identifiers of all nodes along a path from a root node to said node to be inserted.

11. (Previously Presented) The article of manufacture of claim 10, wherein when said node to be inserted is inserted between two existing siblings under said single parent node and when the

string length of local identifier of said first sibling is less than the string length of the local identifier of said second sibling, comprises modules implementing code to:

a. check when local identifier of said first sibling is the last available encoding value having a string length of the local identifier of said first sibling and being smaller in value than said local identifier of said second sibling,

b. when said local identifier of said first sibling is the last combination having a string length of the local identifier of said first sibling that is smaller in value than said local identifier of said second sibling,

i. when the local identifier of said second sibling is not the first available identifier having the string length of the local identifier of said second sibling that is greater than the value of said local identifier of said first sibling; an even-valued local identifier being less in value than said local identifier of said second sibling and having string length of local identifier of said second sibling is generated and assigned, else

ii. generate a local identifier for said node to be inserted by replacing said last byte of said existing first child by an odd number and extending local identifier of said existing first child by a byte having a value of any arbitrary even number less in value than said last byte of said existing first child, and

generate a node identifier by a concatenation of local identifiers of all nodes along a path from a root node to said node to be inserted.

12. (Previously Presented) The article of manufacture of claim 10, wherein when said node to be inserted is inserted between two existing siblings under said single parent node and when the

string length of the local identifier of said first sibling is equal to the string length of the local identifier of said second sibling, comprises modules implementing code to:

a. when the value of the local identifier of said first sibling plus two is less than the value of the local identifier of said second sibling, a local identifier for said node to be inserted takes on an even value greater than or equal to the value of said local identifier of first sibling plus two and less than the value of the local identifier of said second sibling,

b. when the string length of the local identifier of said first sibling plus two is equal to the string length of the local identifier of said second sibling, then the string length of the local identifier for said node to be inserted is extended wherein the length of the local identifier for the newly inserted node is the string length of said second sibling plus one, and the value of the first string length of said first sibling bytes is the node identifier of said first sibling plus one, and the new byte is an arbitrary even number less than the value of said last byte of the node identifier of said second sibling, and

generate a node identifier by a concatenation of local identifiers of all nodes along a path from a root node to said node to be inserted.

13. (Previously Presented) The article of manufacture of claim 10, wherein when said node to be inserted is inserted between two existing siblings under said single parent node and when the string length of the local identifier of said first sibling is greater than the string length of the local identifier of said second sibling, comprises modules implementing code to:

a. when the local identifier of said second sibling is not the smallest value having the string length of said second sibling that is greater in value than the local identifier of said first sibling, then a local identifier having a string length of said second sibling and having even value

smaller than the value of the last byte of the node identifier of said second sibling is generated and assigned else,

b. when the local identifier of said first sibling is not the largest value with the string length of the local identifier of said first sibling, one of the larger values for the new encoding is generated and assigned, else

c. extending the local identifier of said first sibling by a length, by setting the last byte to the highest odd number and the new byte to an even number less than the value of the last byte, and

generate a node identifier by a concatenation of local identifiers of all nodes along a path from a root node to said node to be inserted.

14. (Previously Presented) The computer-based method of claim 1, wherein said assigned local identifiers are assigned values based on variable-length binary string encoding.

15. (Previously Presented) The article of manufacture of claim 9, wherein said assigned local identifiers are assigned values based on variable-length binary string encoding.

16. (Previously Presented) A computer-based method comprising the steps of:

a. choosing an initial base length with which to encode local identifiers,

b. assigning a value of zero as a node identifier to a root node in a logical tree,

c. sequentially assigning to descendants of a root node a local identifier having an even value and a length equal to said base length chosen in said choosing step, wherein said local identifiers are assigned said even values based on variable-length binary string encoding and said local identifiers are assigned in increasing value from leftmost children to rightmost children,

d. assigning node identifiers by concatenating local identifiers of all nodes along a path from a root node to a node to which a node identifier is currently being assigned, and

extending said initial base length when local identifier encoding combinations are exhausted before all descendants are assigned local identifiers.

Evidence Appendix

None

Related Proceedings Appendix

None